CybFIMC - User Manual

Fuzzy-model-based control for the process industry applications

PROCESS
CYBERNETICS
LIMITED

Published and distributed by Process Cybernetics Limited

Disclaimer

Process Cybernetics Limited makes no representation or warranties with respect to the programs or manual and specifically disclaim any expressed or implied warranties of merchantability or fitness for purpose. In no event shall Process Cybernetics Ltd be held liable for direct, indirect, consequential or incidental damages resulting from any defect or omission in the manual, or other related items or processes, including, but not limited to , any interruption of service, loss of business, income, anticipated profit, use, information, or any other consequential damages.

Trademarks

Table of Contents

0

Introduction

Welcome to CybFIMC. CybFIMC is a fuzzy model based controller for Multi-Input/Single-Output (MISO) processes. CybFIMC integrates easily with your existing Windows based SCADA system through Dynamic Data Exchange(DDE) links. All I/o is handled by your SCADA system, and CybFIMC simply adds extra functionality.

CybFIMC uses controller definition files generated by CybMod, Process Cybernetics' fuzzy model building package. These definitions contain a fuzzy model of the process to be controlled, along with controller tuning information.

CybFIMC is based on leading edge control algorithms developed exclusively by Process Cybernetics. CybFIMC is the first true fuzzy-model-based controller available commercially anywhere in the World.

We have developed CybFIMC from leading edge research in fuzzy modelling and control in the process industries. Process engineers have designed the package with the process industries in mind. We are continually developing our products, and would be delighted to receive your feedback.

System Requirements

Pentium PC running Microsoft Windows 95, 98, or NT, and a SCADA systems capable of supporting DDE links.

Installing CybFIMC

Insert the diskette number 1 into the floppy disc drive. Choose the **Run** command from the **Start** menu, and in the Run window type:

a:\setup.exe

and then press enter.

Follow the on-screen instructions to complete the installation.

Verifying CybFIMC

The first time you run CybFIMC you will be taken to a licence verification dialogue. Simply enter your password and press the 'OK' button, then follow the on-screen instructions.

Each copy of CybFIMC is licensed to run on only a single machine. The number of installations is strictly limited by the protection software, and these installations are only available to help you in the event of a hardware problem. If you exceed the number of permitted installations you should contact Process Cybernetics. Process Cybernetics reserves the right to charge for, or refuse to supply, additional registrations.

1

How does CybFIMC work?

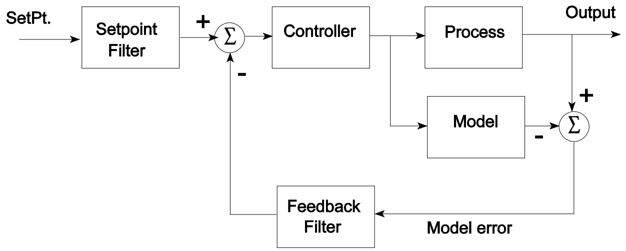
To get the most out of CybFIMC it is important to understand a little of how the controller works. This chapter gives you this information, and also explains how to build models which are suitable for control purposes.

Fuzzy Internal Model Control (FIMC)

CybFIMC is a fuzzy internal model controller. This type of controller was first developed at the University of Strathclyde in Glasgow in 1997.

FIMC is the fuzzy, non-linear, version of the well-known linear IMC controller. The way the controller operates is best understood by looking at the diagram:

The setpoint, or desired value, for the controlled output is fed through a filter and comparator into the controller. The setpoint filter is used to 'shape' the systems response to setpoint changes, and is one of the ways of tuning the controller.



The controller uses the difference between the filtered setpoint and filtered model error to calculate an appropriate control action to apply to the process. The controller is actually just an inverse of part of the process model.

The output from the controller is fed to the process (through the SCADA system) and simultaneously to a copy of the process model. The outputs of the process (a measurement from the real process) and the model (what the model thinks the process output should be) are compared and fed back as the 'model error'. This model error is actually made up of two components: the real error in the model prediction; and the effect of any unaccounted for disturbances on the process output.

The model error is then fed-back into the controller through a filter which can be adjusted to alter the controllers behaviour.

The controller, model, two comparator and two filter blocks are all contained within CybFIMC.

The model within CybFIMC is built using CybMod, Process Cybernetics' model building package. Dead-time on process manipulations will automatically be compensated for due to CybFIMC's controller structure. Also, if disturbance variables are included in the model, then CybFIMC will automatically provide dynamic feedforward action to regulate against them.

Building models for control

The model in an IMC controller is obviously of critical importance in achieving good performance from the controller. Models for CybFIMC are built using the model building package CybMod. Although building models for control purposes is

quite easy, there are a few points which have to be borne in mind.

Incomplete models

The model must be complete. Incomplete models are models containing areas where no identification data was available. This results in 'holes' in the model where predictions cannot be made. These models cannot be used for control, and CybMod will generate an error message if you try to use such an incomplete model to produce a controller definition file.

Incomplete models can be fixed in three ways:

a) Gather data in the incomplete regions. In practical situations this is usually rather difficult to do since there is often a good reason for the data being missing in the first place. It is, however, the best solution if it can be achieved.

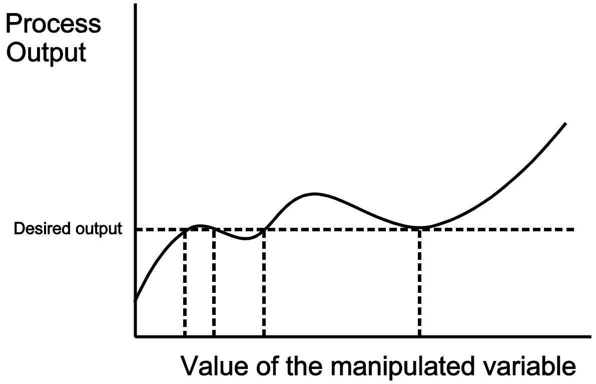
b) Reposition the reference sets which are causing the problems. You should be able to see from the completeness dialogue which variables/sets are causing the problem.

c) Reduce the number of sets on one or more of the model variables.

Non-Monotonic models

A non-monotonic model is one whose prediction surface has folds and bumps along the manipulated variable axis. The effect of a non-monotonic model can easily be understood by considering the graph below

You can see that for a particular value of desired process output there are several potential values of the manipulated variable, all apparently equally valid. With a prediction surface like this CybFIMC would be unable to uniquely invert the process model, and would be incapable of generating an output.



CybMod automatically checks the prediction surface along the manipulated variable axis, and generates an error if you try to use a non-monotonic surface to generate a controller definition file.

The most common way of producing a non-monotonic model is to use too many reference sets to describe the manipulation and then identify the model using data with noise present.

The easiest way to solve non-monotonic problems is to reduce the number of reference sets used to describe the manipulated variable. The stacked surface plots in CybMod are a useful tool in diagnosing this problem.

Model order

CybMod forces you to use first order models to generate controller definition files. (i.e. models with a single lagged value of the output on the right-hand side, and only one lagged value of each of the other inputs).

A rule of thumb in linear model identification is that pessimists believe that only first-order models can be successfully identified from process data, and that optimists think second-order models are OK too. When the extra complications of non-linear modelling are added, the pessimists view point has to win out, and this is a design decision we have made when coding CybFIMC.

CybMod will check the order of any model used to generate a control definition file and will generate an error if the model order is higher than 1.

Tuning CybFIMC

Tuning CybFIMC is simple compared to tuning even a PID controller. Tuning is achieved by altering the behaviour of the setpoint and feedback filters.

Tuning the Setpoint filter

The setpoint filter is used to limit the speed of response of CybFIMC to setpoint changes. If a filter constant of one is used, then CybFIMC will try to drive the process output to the setpoint at the next sample interval. If a filter constant of zero is used then CybFIMC will completely ignore setpoint changes.

A rough idea of the speed of response can be obtained using the formula

$$\text{Time for complete change} = \frac{3 \times \text{Sample period}}{\text{Filter constant}}$$

The only use for the setpoint filter is to limit excessive movement in the manipulated variable on sudden changes in setpoint. The value of the

setpoint filter has no effect on disturbance rejection, nor on the stability of the controlled response.

Tuning the feedback filter

The feedback filter can be altered in two ways. By selecting the 'Derivative action on feedback' in the controller definition dialogue of CybMod a first-order filter is used to filter the error feedback. If the box in the dialogue is left unchecked then a second-order error filter is used.

A first-order error filter will lead to derivative type action in CybFIMC's error response. This will produce the fastest response to errors or disturbances, but can run into problems in fast, noisy systems. The use of a second-order filter (leaving the box unchecked) will result in a slower overall response, but one which is relatively noise tolerant.

The feedback filter constant works in exactly the same way as the setpoint filter. Selecting a value of one results in the fastest possible response. Selecting a value of zero switches off error feedback, converting CybFIMC into a purely feedforward controller.

Tuning the feedback filter is more difficult than tuning the setpoint filter. Stability of the overall control system can be affected by the value of the feedback filter because it sits within the controllers feedback path. The degree of stability problems depends on how closely the process model matches the real process behaviour. With a perfect model (which is highly unlikely), and an open-loop stable process, the filter constant can be set at values close to one providing the fastest possible response. As the model accuracy declines, then the filter constant needs to be moved away from one to maintain stability, and this reduces the speed of response. Although a bit of trial and error is involved, the tuning process is really not very difficult (much simpler than tuning a PID controller).

2

Linking to a SCADA System

CybFIMC is designed to be simple to integrate with your existing SCADA package. This chapter provides you with the general approach which can be used, but the details will be dependent on your particular SCADA software.

Dynamic Data Exchange (DDE)

Dynamic Data Exchange is a method, available in the Microsoft Windows operating systems, for exchanging data between programs. DDE works by using an area of shared memory. It is a bit like using the 'clipboard' to transfer information between programs but occurs completely under program control, without user intervention.

CybFIMC is a *DDE server* which means that it acts primarily as an information source. To connect to CybFIMC your SCADA software must be able to act as a *DDE client*. Consult your SCADA documentation to confirm this.

The key to allowing data to be exchanged through DDE is the *DDE address* of the information to be connected to. The DDE address is made up of three components: the application name; the topic name; and the item name.

The application name is the name of the server which is to be used to supply, or receive, the data. This will always be 'CYBFIMC' when the SCADA system and CybFIMC are running on the same machine (it is not recommended to run the two pieces of software on different machines over a network).

The topic name is generally used to indicate a particular group of data. In CybFIMC the topic name is the name you used when configuring the controller file in CybMod. If you plan to run multiple copies of CybFIMC, it is important to ensure that each controller name is unique.

The item name is the name which connects to a particular piece of data. CybFIMC generates item addresses automatically and lists them against the data field tag names on the CybOnLine faceplate.

Most SCADA systems will allow DDE servers to act as data I/O sources for a wide variety of input and output variable types (eg analogue outputs, analogue inputs, digital outputs and digital inputs). The usual way to set up a link is to specify DDE as the 'I/O device', and then to include the DDE address as a device I/O address. The method of forming the individual components of the DDE address (application, topic and item) into a single I/O address varies between SCADA systems - consult your SCADA documentation to find out how to do this.

Triggering the calculation

CybFIMC provides you with a choice of ways to trigger a calculation:

Automatic triggering

In this mode CybFIMC keeps track of input information which has been passed from the SCADA package. A control calculation is triggered when a complete set of input data has been received. Once a calculation has been made, all of the input counters are set to zero, and a new, and complete, set of input data has to be sent to trigger a new calculation.

External Triggering

When running in this mode a control calculation is only triggered when a signal is send to a specific DDE trigger address. This would normally be achieved by creating a triggering output in the SCADA system.

CybFIMC still keeps track of the inputs which have been received and will generate an error if the same input is received twice before a trigger, or if an input is not received at all before a trigger arrives.

The choice of automatic or external triggering is yours. Depending on the SCADA system being used a particular method may be easier to implement.

Running CybFIMC

CybFIMC must be running before your SCADA system starts to scan. If it isn't running then the SCADA system will try to pick up DDE addresses in CybFIMC which don't exist and scanning errors will result.

There are two ways of ensuring that CybFIMC starts in time:

Adding CybFIMC to the Windows 'StartUp' Folder

Windows includes a special 'StartUp' folder. Programs contained in this folder will automatically be started immediately after the operating system starts. Your SCADA system installation has probably put a start-up component in this folder to automatically start your SCADA software.

To add CybFIMC to this start-up group do the following:

1. Click on the start button, and then point to Settings

2. Click 'Taskbar', and then click the start menu programs tab.

3. Click Add, and then type something like the following in the command line edit box.

```
"c:\program files\cybfimc\cybfimc" /t "c:\tmp\cont.imc"
```

The directory 'cybfimc' may be different if you have chosen to install CybFIMC to a different directory. The '/t' switch instructs CybFIMC to use external triggering. If you want CybFIMC to automatically trigger calculations then simply omit the switch. The 'c:\tmp\cont.imc' is the controller definition file that you wish to load into CybFIMC. This definition file must have been previously created in CybMod. Remember to include the path name to allow

Windows to find the file.

4. Click the 'Next' button and then click the 'StartUp' folder.

5. Type the name you want to see appear in the Start-up menu and then press the 'Finish' button. The name won't affect the programs operation and is really for your own information.

Multiple copies of CybFIMC can be run simultaneously. Simply add more start-up command lines to the 'StartUp' folder, remembering to use different controller definition files for each.

Adding CybFIMC to the SCADA software's start-up list

Most SCADA packages include the facility of automatically starting additional software on SCADA start-up. The details of how to do this will be in your SCADA packages documentation. Remember that you will need to be able to edit the command line for CybFIMC to allow the controller definition file and trigger switch to be specified.

Again multiple copies of CybFIMC can be run simultaneously by including multiple start-up command lines.

The CybFIMC faceplate and linking to the SCADA system

When CybFIMC is running you will see an application Window containing a number of 'cards' marked 'Manipulation', 'Comms Status' and 'Inputs'. Each card contains important information to help you link your SCADA system to CybFIMC. To access any card simply click the tab at the top of the card.

Right at the top of the CybFIMC Window you will find two fields labelled 'DDE Application Name' and 'DDE Topic Name'. These fields provide you with the first two parts of the DDE address (Application and topic name) for <u>all</u> of the other fields in this instance of CybFIMC.

The 'Inputs' card gives details of the DDE addresses associated with the controller inputs. The first column in the card lists the tagnames of the variables associated with the controller. As well as current values of the inputs to the model you created in CybMod, the current output value is required too. Another variable, the controller setpoint, is also required. This has a tagname which is created by adding 'SP' to the end of the controlled output tagname. The next column gives the DDE item name for each of the controller inputs. This item name, in combination with the general application and topic names, forms the complete DDE address for the variable. The last column shows the current value of the particular variable, and is for information only.

The 'Comms Status' card is used to provide information on the DDE communications. On this card a DDE item name is given for an error flag variable. This flag variable provides a way to warn operators of problems with DDE communications to CybFIMC. When there are no problems this flag variable has a value of zero, which changes to 1 on a communication error. A 'Reset' button is provided on this page to reset the controller after any communication problems.

If external triggering is selected then the 'Comms Status' card also provides the DDE item name of the trigger variable (in the column 'Trigger Item'). With external triggering it is necessary to send something to this address to trigger a calculation (it doesn't matter what is transmitted, it is simply the act of transmission which triggers the calculation).

The 'Manipulation' card provides information related to the controller output. The controller output is the suggested value of the manipulated variable. The DDE item address is provided to allow this to be linked to an appropriate SCADA system variable.

Procedure for linking to CybFIMC using

Automatic Triggering.

1. Build a controller definition file in CybMod, and load into CybFIMC (omit the /t switch on running to disable external triggering).

2. Create an analogue output variable in the SCADA system for each variable listed in the 'Inputs' card in CybFIMC. The I/O addresses are made up of the application, topic and item names given on the card. The setpoint variable will probably have to be supported by additional SCADA programming to allow the operators to change it. The sampling time for the variables should be set to be the same as the sampling time used in the data which was used to identify the controller model.

3. Create a digital input in the SCADA system and connect it to the error flag on the 'Comms Status' page. Add extra alarm logic in the SCADA system to warn the operators of a communication problem. Set the sampling period to be the same as for the variables in (2), but offset the sample by a few seconds to give the calculation time to complete.
4. Create an analogue input and connect it to the DDE address given in the 'Manipulation' card. Additional SCADA logic is then required to connect this variable with the process. It is possible to just chain this input to an analogue output and then straight through to a valve, but

it is much more common to use the value as a setpoint for a PID type flow controller. The sample time on the analogue input should be the same as the variables in (2), but should be offset by a few seconds to give the calculation time to complete.

Using external triggering

Most of the procedure above is the same when an external trigger is used, but there are the following differences:

1. CybFIMC is started with the '/t' switch included in the command line.

2. Another analogue output variable, the trigger variable, needs to be created in the SCADA system. This variable should point to the DDE address indicated in the 'Comms Status' card. The sample time should be the same as for the other analogue outputs, but should be offset to be the last output transmitted to CybFIMC.

3. The digital (error flag) and analogue (manipulation) input scans to the SCADA system should be offset a few seconds in sample time from the trigger variable to give the control calculation time to take place.

3

Other Services from Process Cybernetics

We hope that you find CybFIMC a useful and easy to use piece of software. All our software is written in-house and we are continually developing our products. We are very interested in receiving your feedback on our products. If you have any comments to make, or suggestions for improvements to our products, then please get in touch with us at the address given at the end of this chapter.

The rest of this chapter outlines some of the other products and services available from Process Cybernetics.

Other products

CybMod - A fuzzy modelling program for process control

CybMod is the core of our current range of products. It generates relational fuzzy models from process input/output data and saves them in a form suitable for use in CybOnLine and CybFIMC.

CybMod includes tools to analyse and adjust the raw I/O data, and model construction and identification is quick and easy. CybMod also includes a range of tools to evaluate the quality of the models, and to diagnose any problems.

CybOnLine - On-line fuzzy modelling

CybOnLine provides an easy way of integrating models developed in CybMod with a wide range of third-party SCADA systems. These models could be used for inferring the values of difficult to measure variables from other measurements, or could be used to provide 'what-if' support for process operators.

CybOnLine is a Dynamic Data Exchange (DDE) server. It runs, as a separate task, alongside your existing DDE-capable SCADA system and communicates via DDE links. These links are easy to set up, usually just by setting an appropriate I/O address.

Each CybOnline module can handle up to nine sub-models (the maximum which can be defined in CybMod), and multiple copies of CybOnLine can be run simultaneously providing as many on-line models as you require.

Services from Process Cybernetics

Training

Process Cybernetics offers courses on various aspects of fuzzy modelling and control. These courses include a mixture of both theory and practise.

If you would like to be kept informed of upcoming courses then tick the boxes on your registration card, or contact us directly at the address given at the end of this chapter.

Consultancy

Process Cybernetics can provide consultancy to help support your modelling and control projects. The level of our consultancy involvement can range from telephone support to full project management.

We shall provide a fixed-price quotation for any work we undertake. If you are interested in our consultancy services, then contact us at the address given at the end of the chapter.

Software services

All our software is written in-house, and, as a result, we are able to offer a 'bespoke tailoring' service to make our software suit your particular requirements.

This could involve changing the look of the software, changing the way it operates, or integrating it with other applications.

If you have an application in mind that requires special software, then contact us at the address at the end of the chapter. We will discuss your requirements and provide you with a fixed price quotation for the work.

Chapter 4